ZUPAN

Take Home Challenge - Mobile developer

Goal: Build a mobile app that can use voice recognition commands and transform them into data that will be shown on the screen.

We use React Native and Typescript but you can use the tech you consider and build a native.

Adding production ready features is **a big plus.**

Adding a secondary language (or anticipating the code to allow for more) **will be a big plus.**
Adding the chance to add new commands later **is a plus.** (open/close code, config)

Please send us a GiHub repository with the code and some instructions to run it.

If you don't have time to implement a production ready feature, please add comments explaining the missing thing and why we should have it.

Example:
Backend API input strings are not validated, we can use a JSON schema library to send error messages to the Frontend.

Speech to command

We want to pick speech and transform it into commands, the commands will have an input described on every command and an output, like { "command" "name", value: "output" }

Commands will start with a keyword followed by some parameters.

For the sake of simplicity, parameters will always be numbers, single ones (so 60 will be six *zero two* instead of sixty), if another word rather than a number is detected, it has to be ignored (**Reset command** is the only exception since it will skip this command and put the module into waiting for command.)

We will have 2 states, waiting for command and listening to the current command.

When the user says a command word, the module will switch from waiting for command to listening to the current command. If there is no command word it will ignore them.

Code command

Input

- Key: Code
- Parameters: Numbers

Output

-

Examples:

- **Count** *four two* => { “command”: “count”, value: 42 }
- **Count** *eleven six* => { “command”: “count”, value: 6 }
    - *This is not a typo since we don’t accept numbers outside the range 0 to 9, so eleven is ignored.*

Count command

Input

- Key: Count
- Parameters: A row of numbers

Output

Example: **Code** *Two Four Five Six* => { “command”: “count”, value: 2456 }

Reset command

Input

- Key: Reset
- Parameters: No parameters

Speech example: Reset
Example: **Reset** => { “command”: “reset”, value: undefined }
Note: This command will cancel current commands

Back command

Input

- Key: Back
- Parameters: No parameters

Speech example: Back
Example: **Back** => { “command”: “back”, value: undefined }

Showing the data

In order to test the voice module we need to show some data.

- Current state of voice module
- Current command if any
- Current parameters if any
- Speech detected in the last few seconds
- Data outputs from commands

When Reset and Back commands, we will need a visual cue to show they were used.
Back command will undo the last data entry.

Let’s say the users says:
Code two three four **Count** sixty two **Code** one two **Reset Code** one one two **Count** five

The final should be:

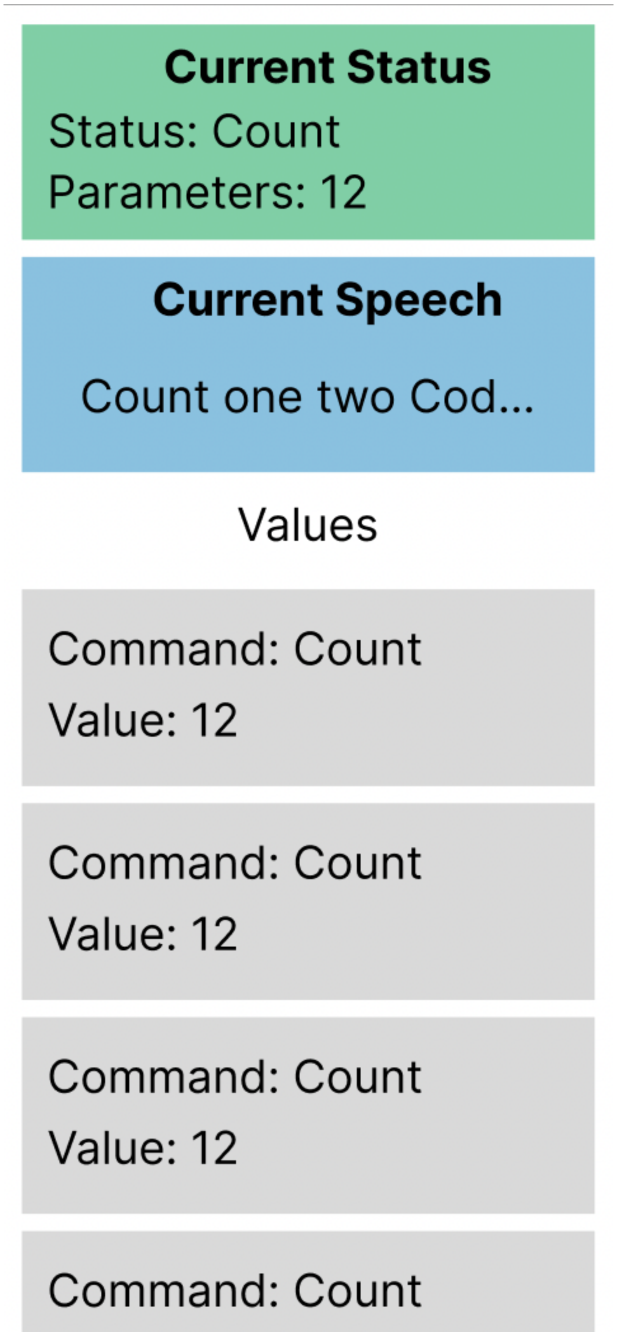
```
{
        “command”: “code”,
        “value”: 234
}
{
        “command”: “count”,
        “value”: 2
}
{
        “command”: “code”,
        “value”: 112
}
{
        “command”: “code”,
        “value”: 5
}
```

Then the users says:
Back

```
{
        “command”: “code”,
        “value”: 234
}
{
        “command”: “count”,
        “value”: 2
}
{
        “command”: “code”,
        “value”: 112
}
```

(Notice we removed the last command)

You can use this image as reference:



Current Status

Status: Count

Parameters: 12

Current Speech

Count one two Cod...

Values

Command: Count

Value: 12

Command: Count

Value: 12

Command: Count

Value: 12

Command: Count